

Event builder components

I) ATM

Brief description
How we use it

II) Hardware

List of components
ATM adapters
VME CPUs
Linux PCs
ATM switch

III) Software

ATM driver for VxWorks
Scanner Manager
Scanner CPU
L3 receiver
L3

IV) Conclusions

ATM
Brief description
How we use it

What is ATM?

Several layers

ATM layer

Fixed-size transmission units (cells)
48 bytes of payload + 5 of routing, etc.
No guaranteed delivery of cells
Each cell belongs to a virtual connection
 Virtual path ID (VPI): set of connections
 Virtual circuit ID (VCI): member of set
Cell transmission order maintained

ATM adaptation layer (AAL)

Fragmentation and reassembly
 Packets to cells and vice versa
Packet CRC and data count (AAL5)
Still no guaranteed delivery

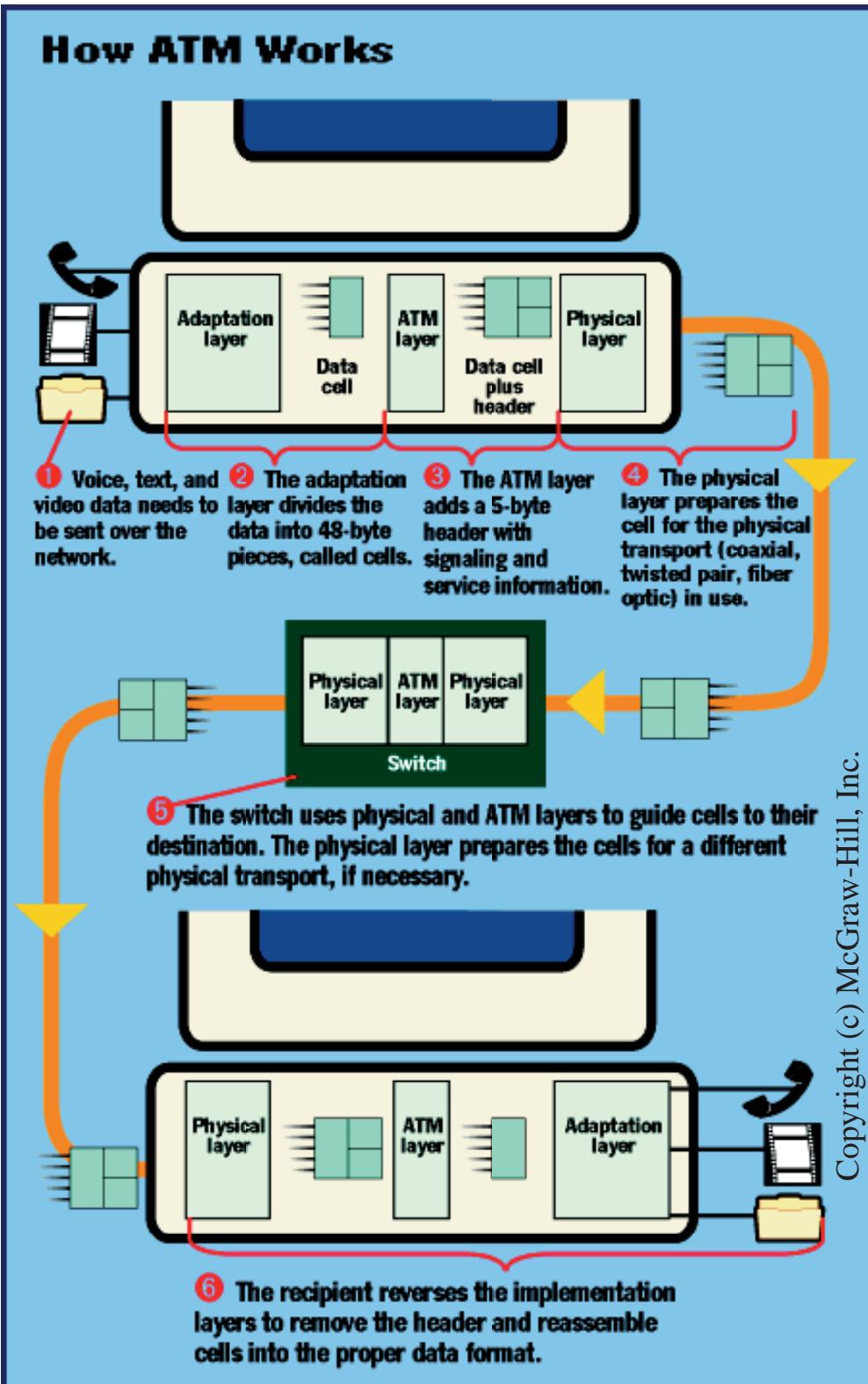
Signalling

Dynamic (switched) virtual circuits
Congestion control

Higher-level protocols

TCP/IP, etc.

ATM data path



What of ATM do we use?

We use

Permanent virtual circuits
A modest amount of traffic shaping
AAL5 packets (64K max payload)

We don't use

Signalling or switched virtual circuits
TCP or other "reliable" protocol (overheads)
Traffic monitoring

We want to keep it simple because

Must port to VxWorks
Local nets can forego "reliable" protocol

Hardware

List of components

ATM adapters

Block diagrams

VME CPUs

Comparison

Block diagrams

Performance

Linux PCs

Block diagram

Performance

ATM switch

Description

Block diagrams

Components

ATM adapters

Interphase 4515 PMC on VME CPUs
ForeRunner LE PCI on PCs

VME CPUs (VxWorks 5.3.1)

Motorola MVME1603 (4) (first generation)
Motorola MVME2603 (10) (2nd gen.)
Motorola MVME2604 (2) (2nd gen.)
PCI with PMC connector
SysTran SCRAMNet VME from Run 1b

Intel PCs (FNAL Red Hat Linux 5)

Cheap commodity boxes
PCI bus
100 Mb/s Ethernet
New SysTran SCRAMNet PCI on one PC

ATM switch

ForeRunner ASX-1000
16 bidirectional 155 Mbps optical ports now
Expandable to 64 155 Mbps ports

ATM adapters

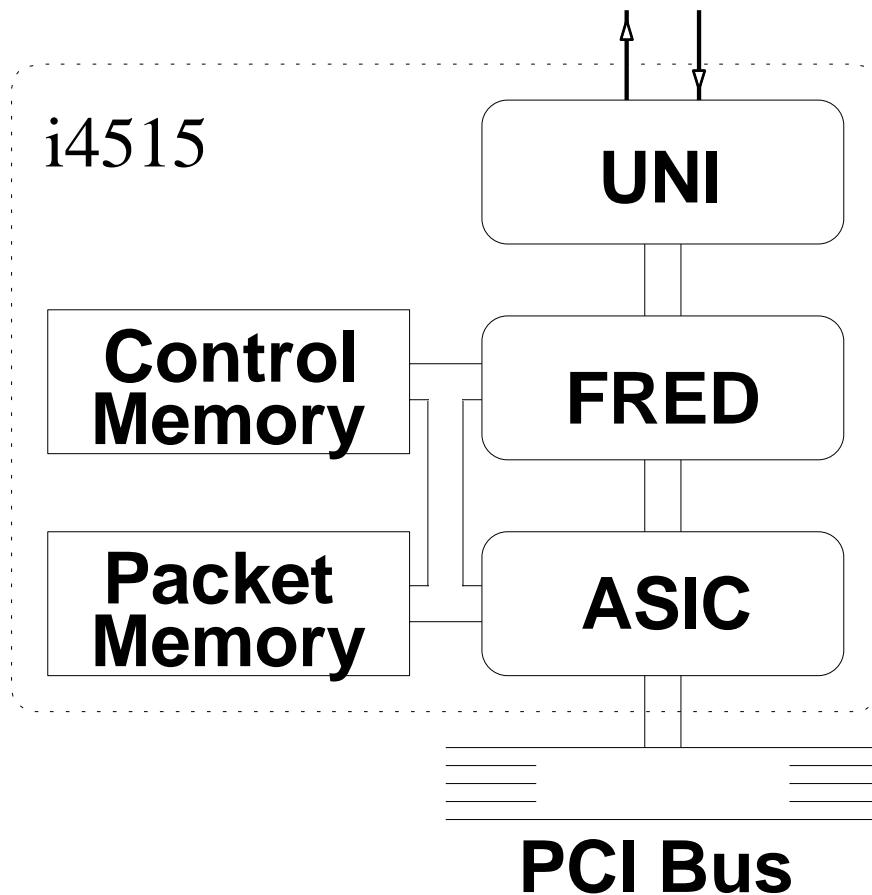
Block diagrams

4515 adapter

Packet assembly in on-board or host memory

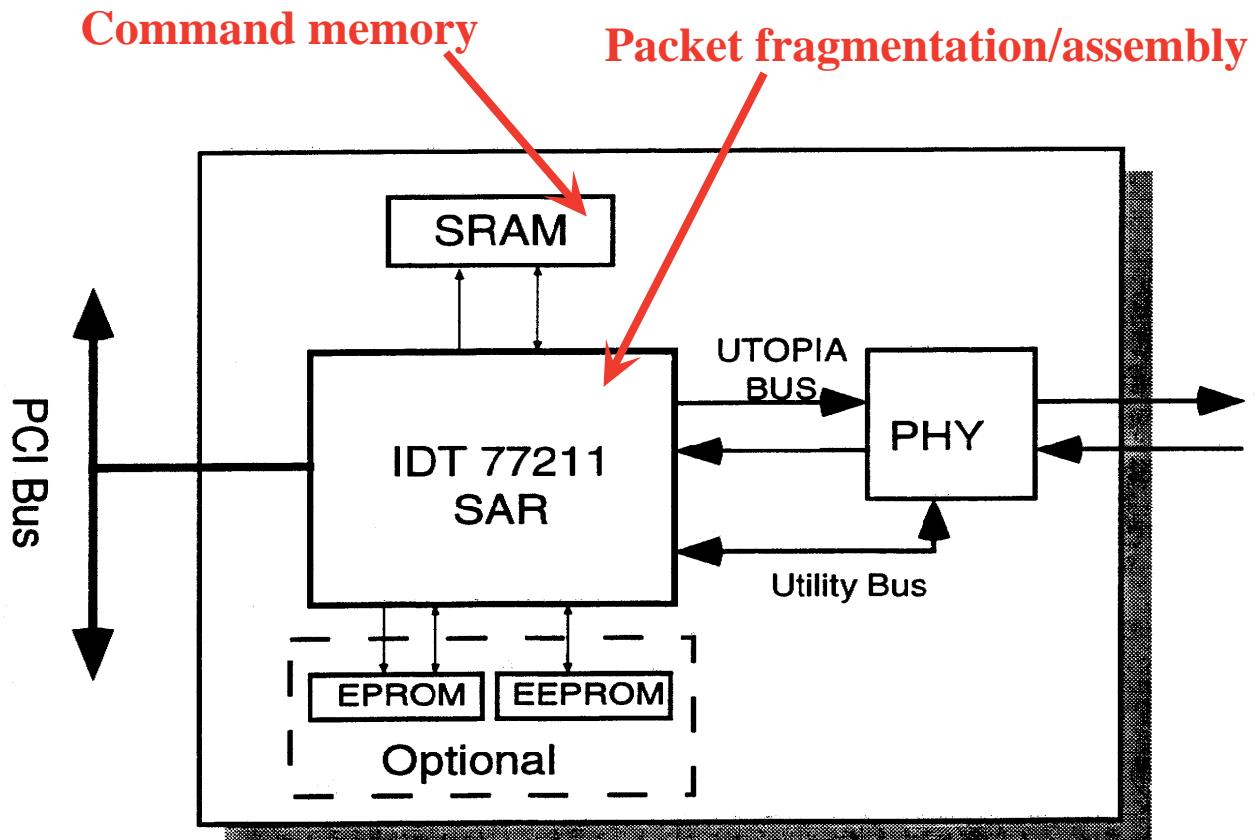
Packet fragmentation from on-board memory

to ATM switch



ForeRunner LE

No on-board packet memory, relies on host



Copyright (c) Fore Systems, Inc.

VME CPUs

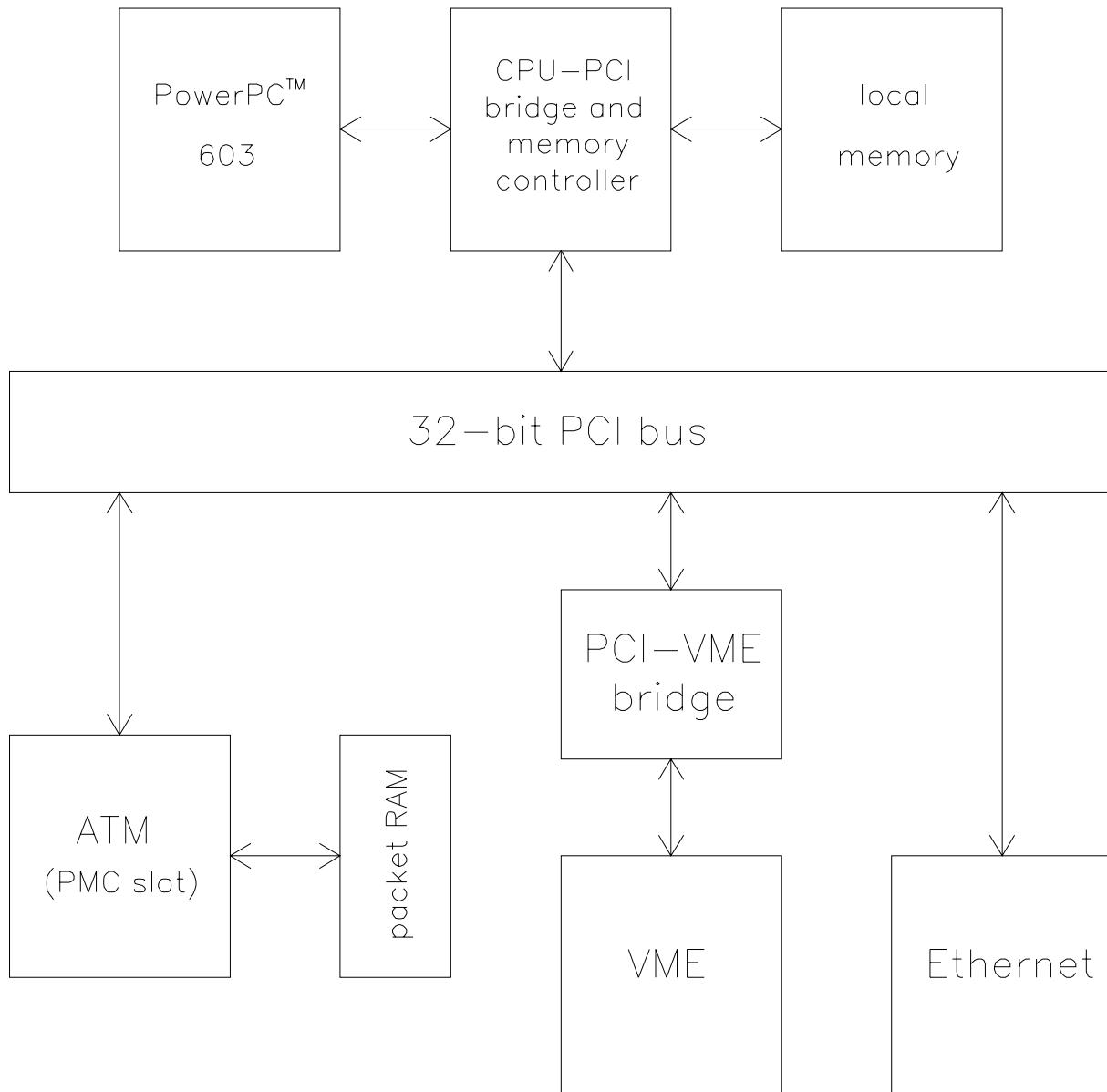
Comparison

Block diagrams

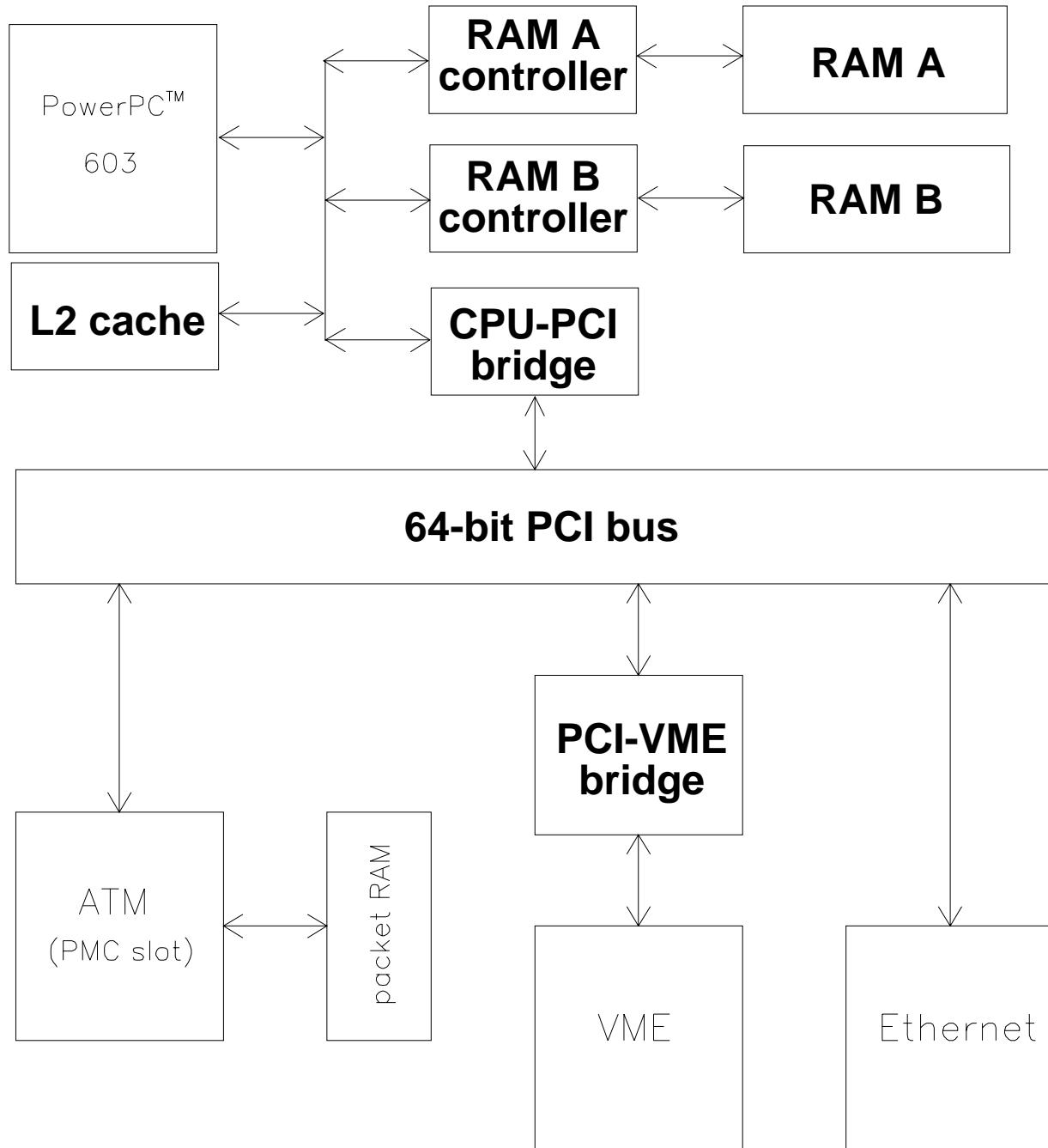
VME CPUs

Component	MVME2603/2604	MVME1603
CPU	PPC 603ev/604e	PPC 603
CPU clock	200 MHz	66 MHz
Memory bus clock	66 MHz	66 MHz
Memory bus width	64 bits	64 bits
Memory controllers	2x64 bits	1x64 bits
Mem. controller chip	Falcon	Eagle
Memory banks	2	1
PCI bus clock	33 MHz	33 MHz
PCI bus width	64 bits	32 bits
PCI-CPU bridge	Raven	Eagle
PCI busses	1	1
PCI (PMC) slots	4x4KB I, ditto D	2x4KB I, ditto D
L1 cache	1	1
L2 cache	256KB unified	None
PCI-VME bridge	Universe	Custom+VMEChip2

MVME1603



MVME2603



VME CPUs Performance

VME CPUs: Memory test

Simple C benchmark

- Fits in L1 cache
- Partially unrolled inner loop
- Reads from an uncached buffer
- 64-bit accesses

Results

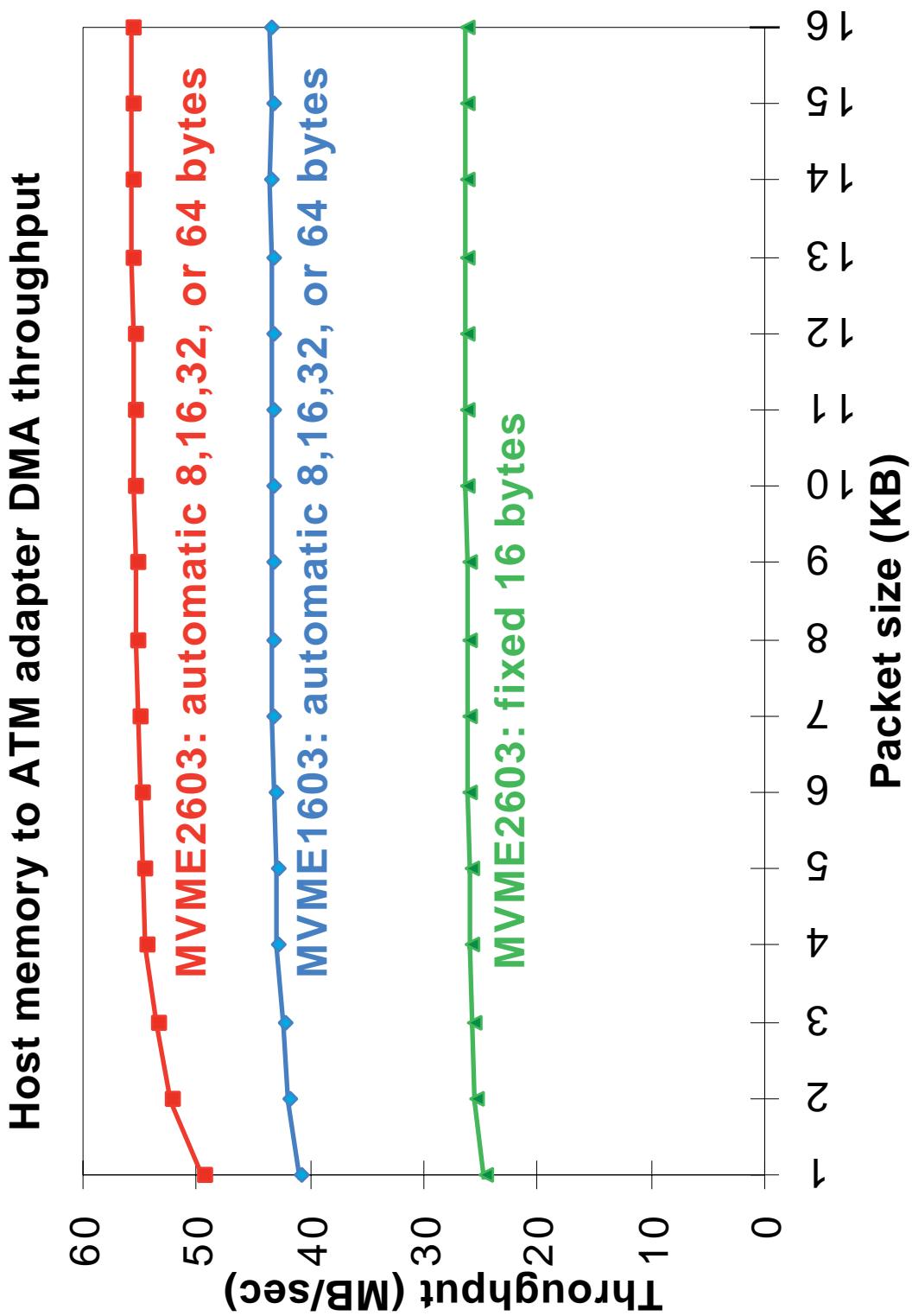
123 MB/s MVME2603

63 MB/s MVME1603

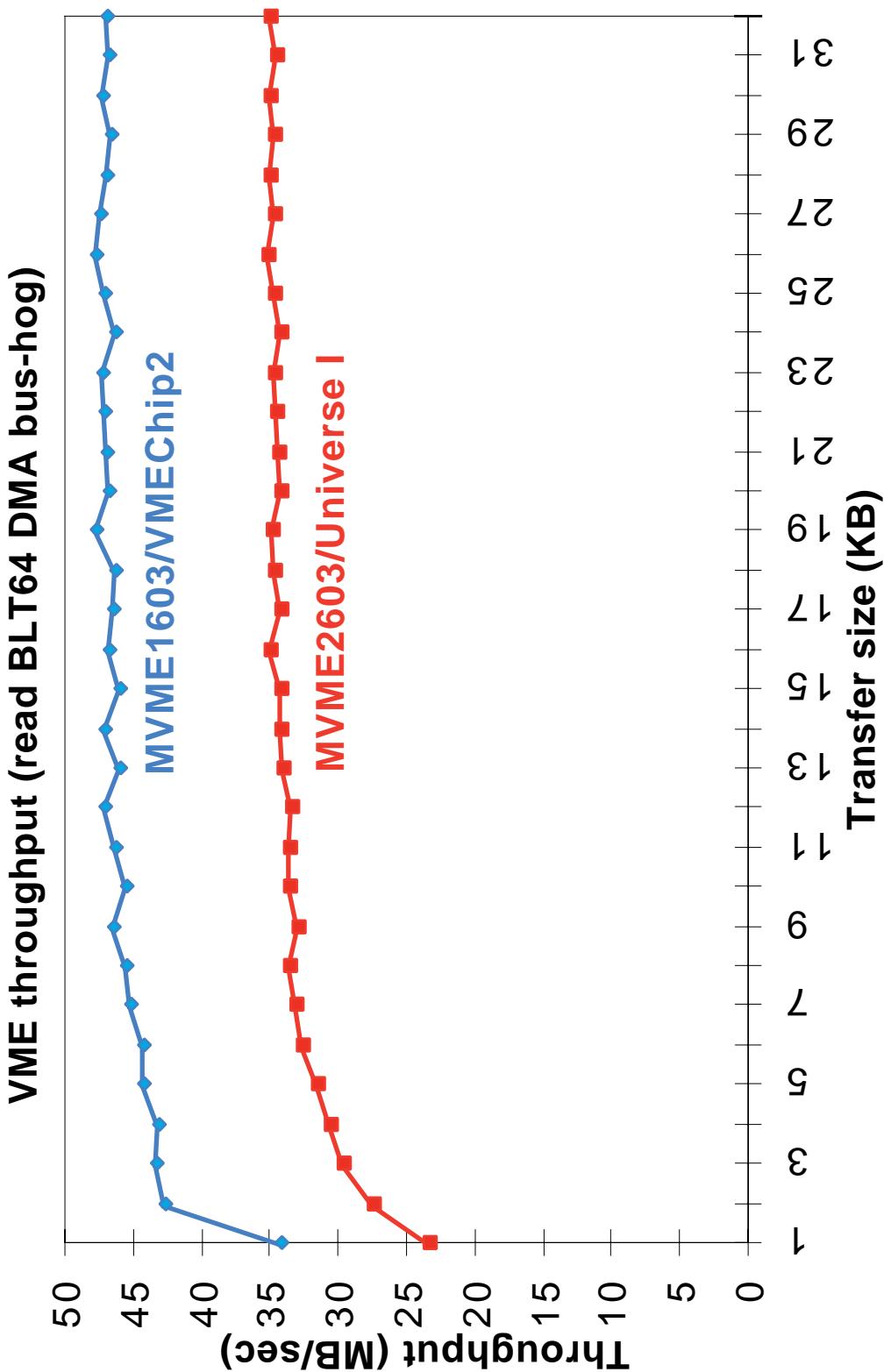
34 MB/s Radstone PPC1-603 rev. 4

**Results proportional to
(memory bus clock) x
(controller path width to memory)**

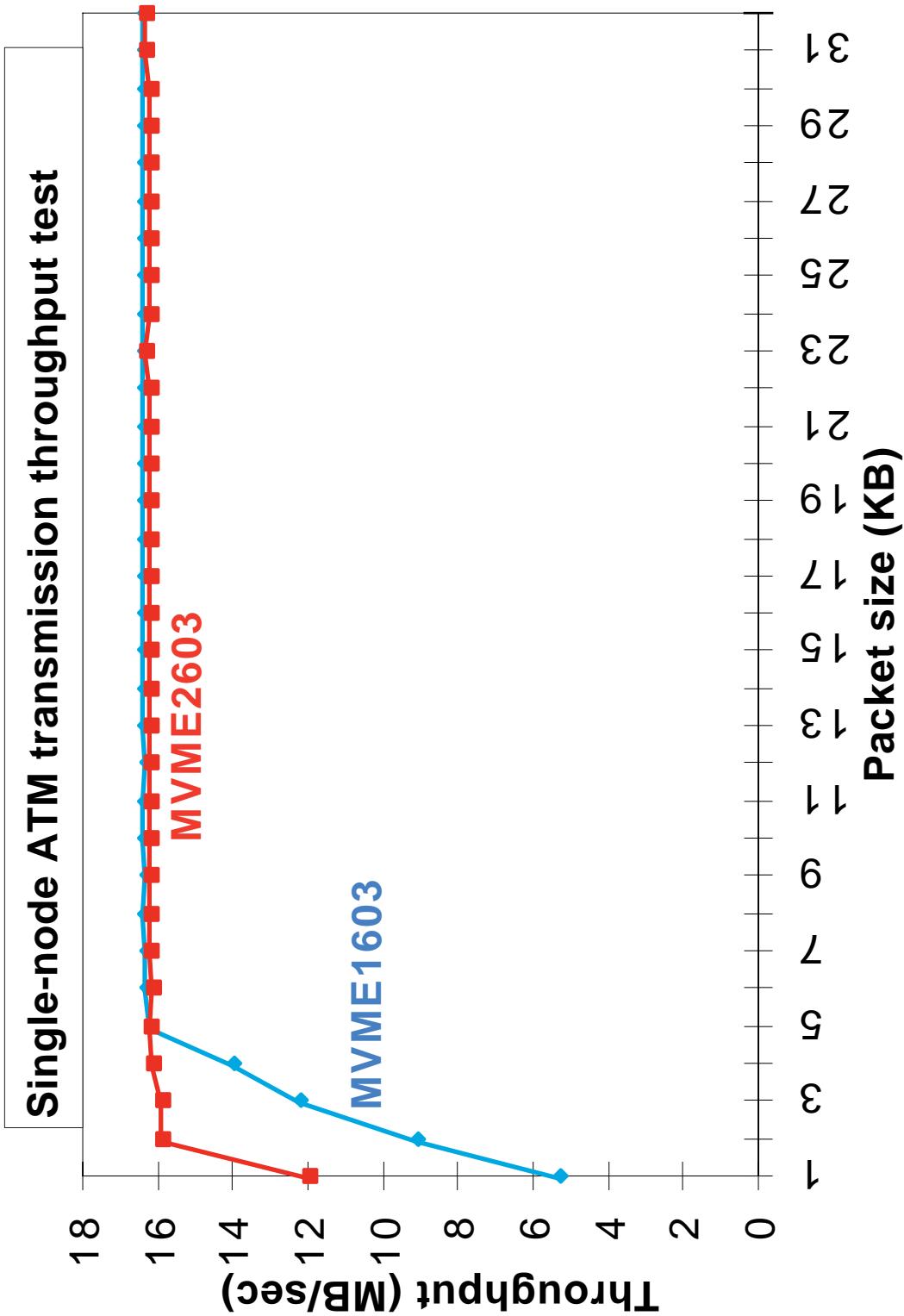
VME CPUs: DMA to ATM



VME CPUs: DMA from VME



VME CPUs: ATM transmission

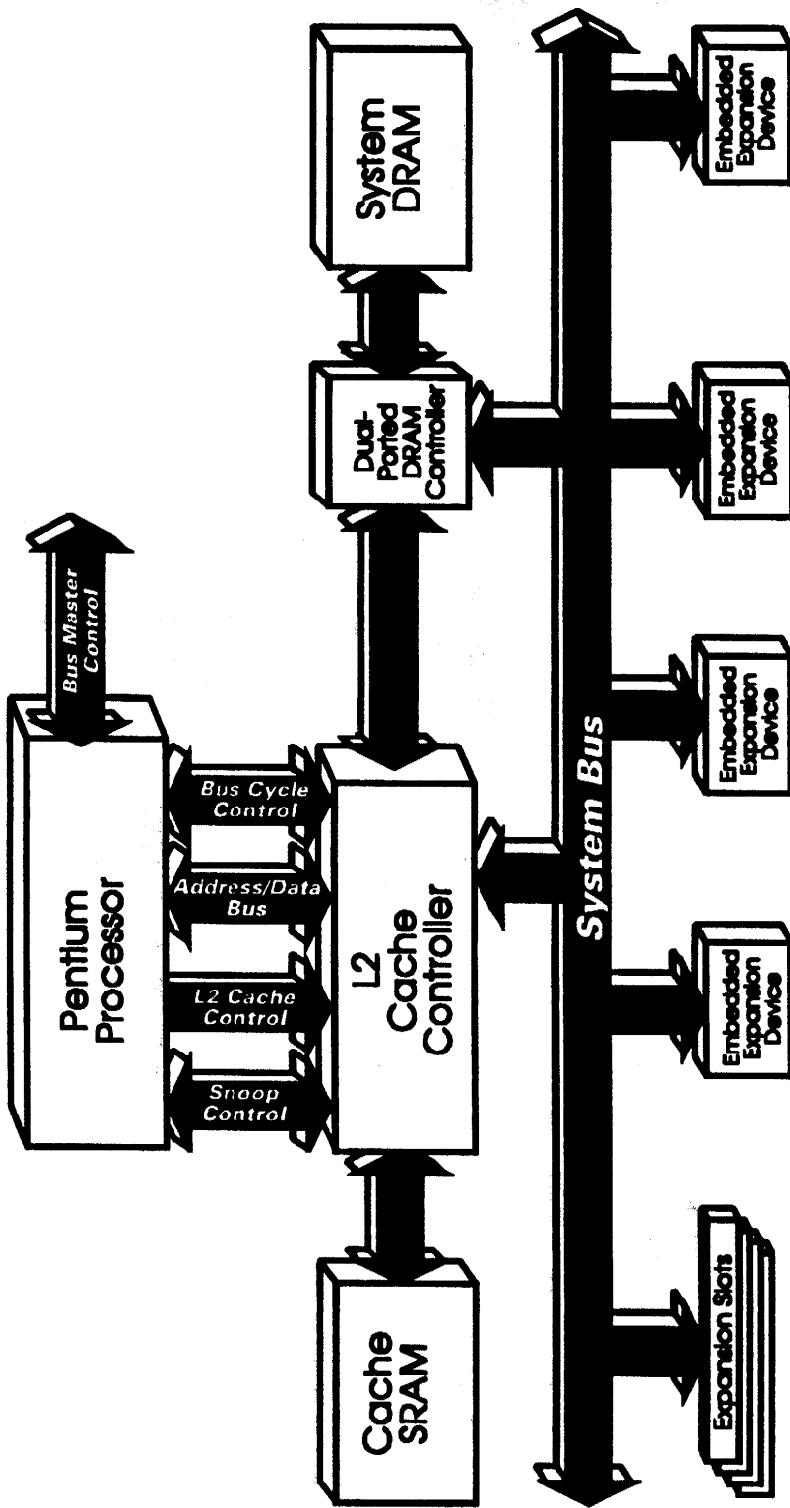


Linux PCs

Block diagram

Performance

PC diagram



Copyright (c) MindShare, Inc.

PC performance & specs

System bus

32-bit PCI

33 MHz clock

32 MB/s ATM DMA tests (memory limited)

CPU

Pentium Pro (dual in compute nodes)

200 MHz clock

RAM

EDO 30 ns (converter), 119 MB/s

FPM 30 ns (compute), 88 MB/s

OS

FNAL version of Red Hat Linux 5

ATM switch
Description
Block diagrams

ATM switch

ForeRunner ASX-1000

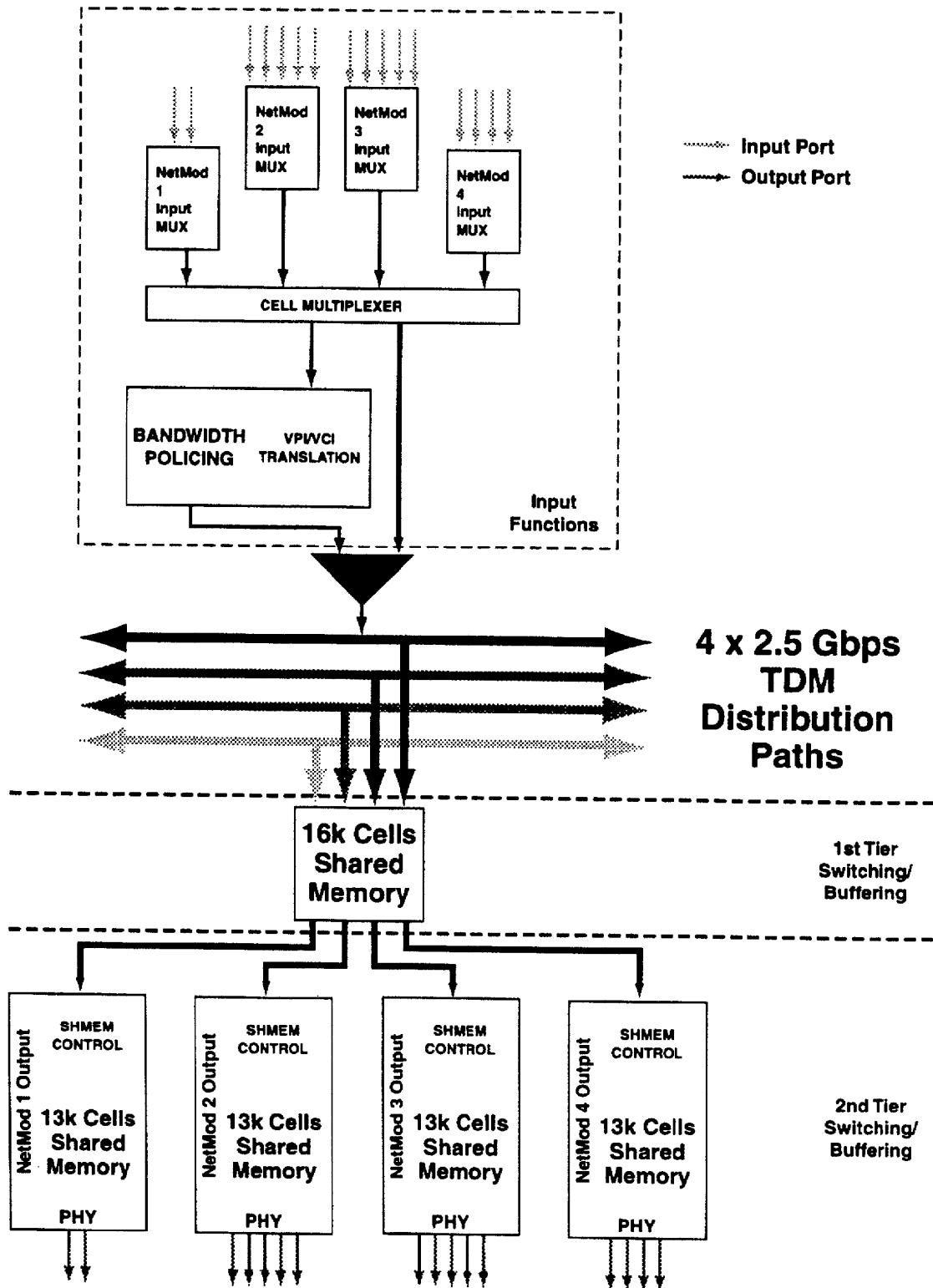
- Non-blocking input
- High internal bandwidth
- Large buffers
- Modular, scalable
- Up to four internal busses (2.56 Gb/s each)

Operating mode

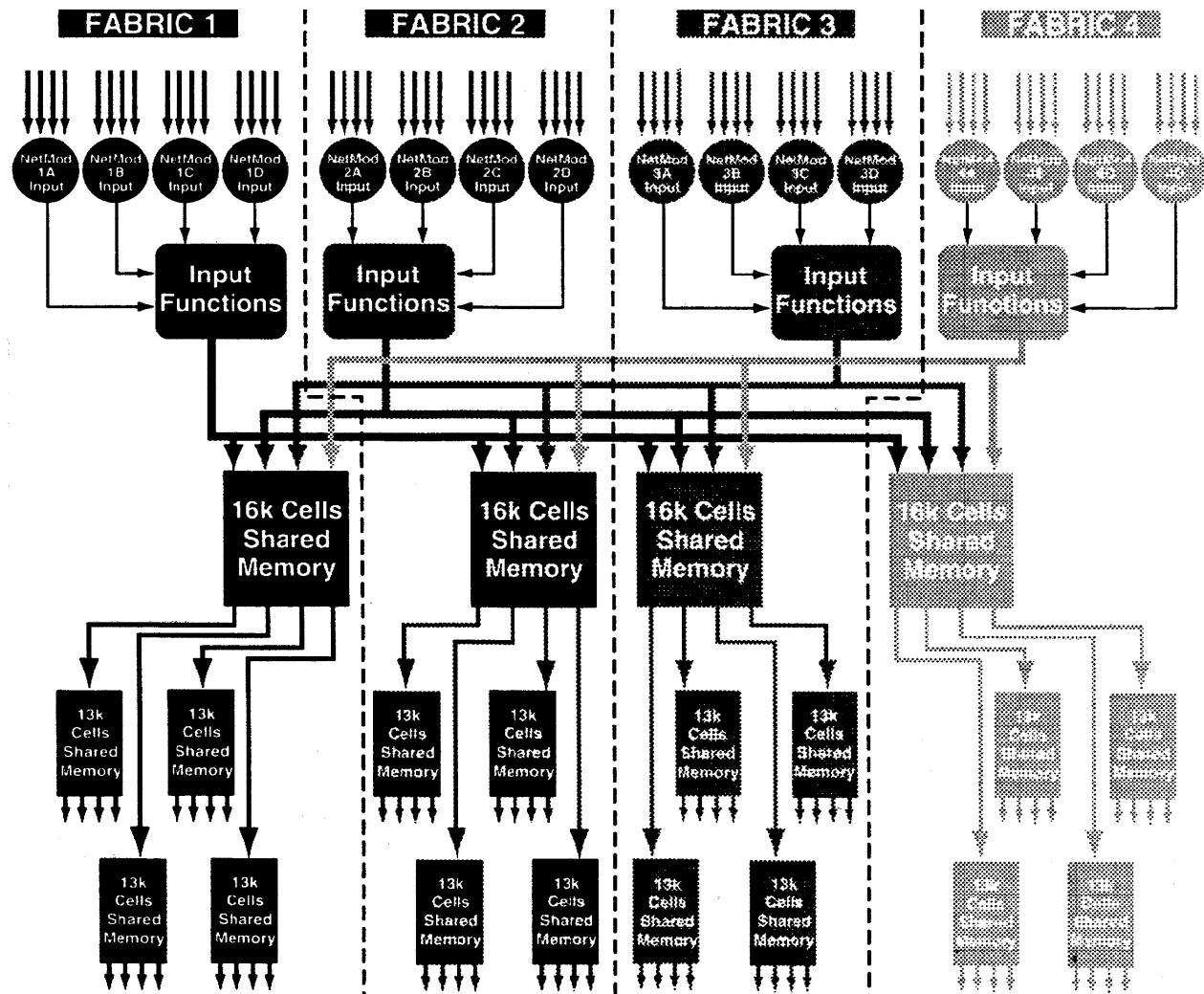
- As "dumb" as possible
- No signalling
- Virtual circuits defined once via telnet
- No traffic monitoring
- Just send what I tell you

Nominal run 2 load is < 20% of one bus

ASX-1000 - one fabric



ASX-1000 - all fabrics



Software

ATM driver for VxWorks
Scanner Manager
Scanner CPU
L3 receiver
L3

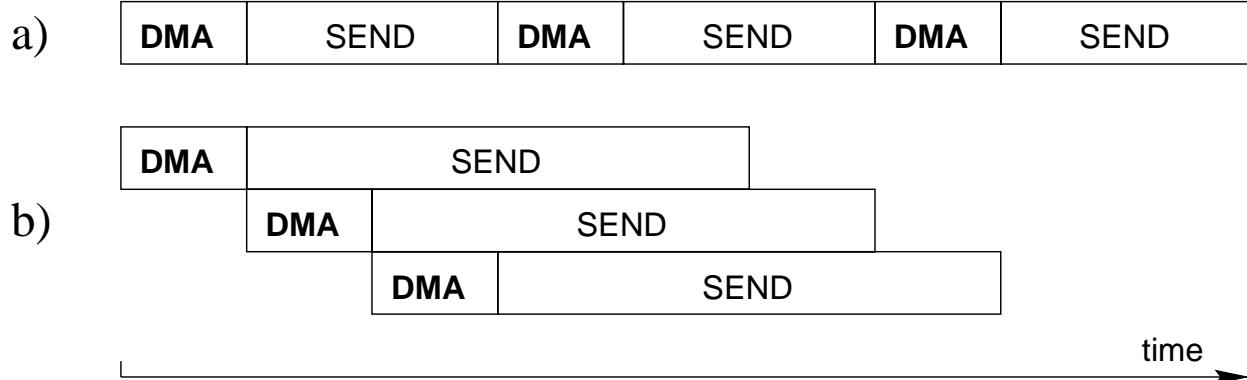
Written here

Modes of operation

(a) Next DMA waits for transmission

(b) DMA overlaps transmission

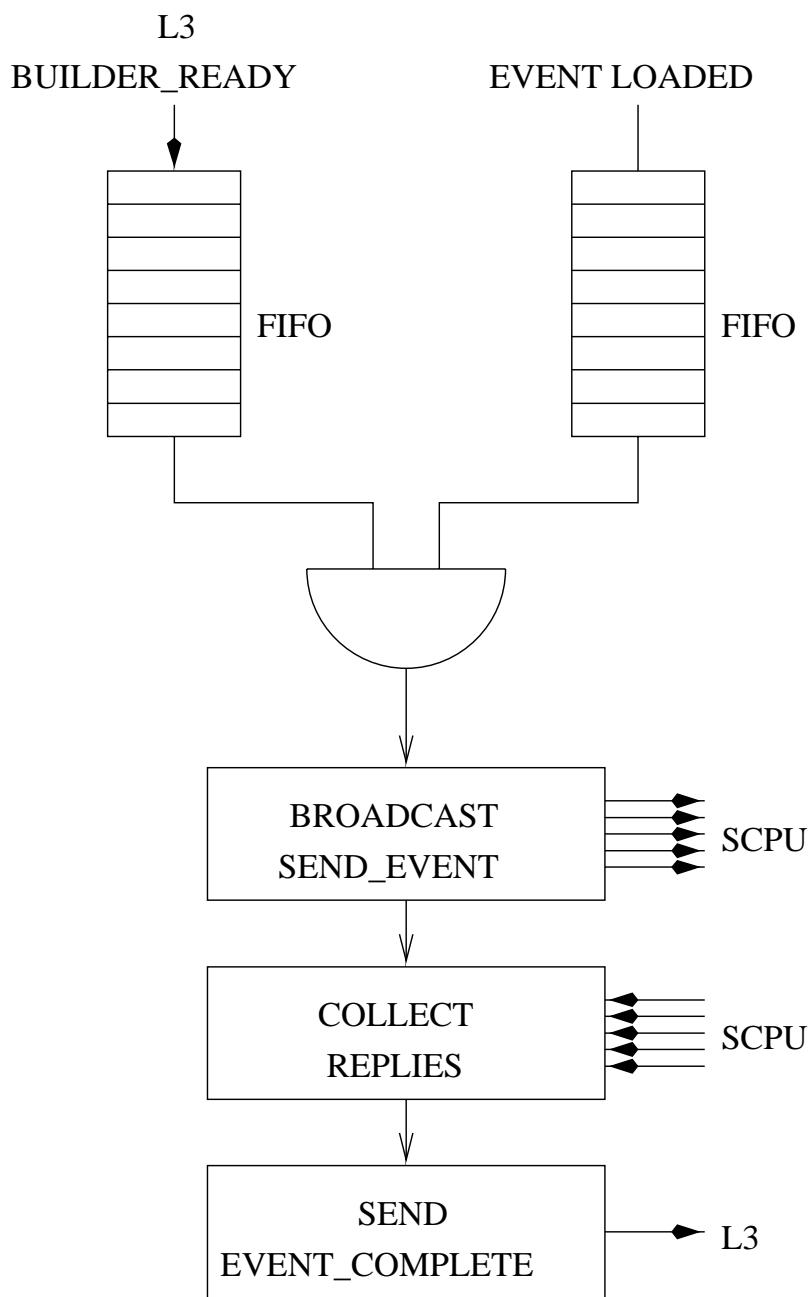
Optional restriction to one active transmission per destination



Scanner Manager code

Role

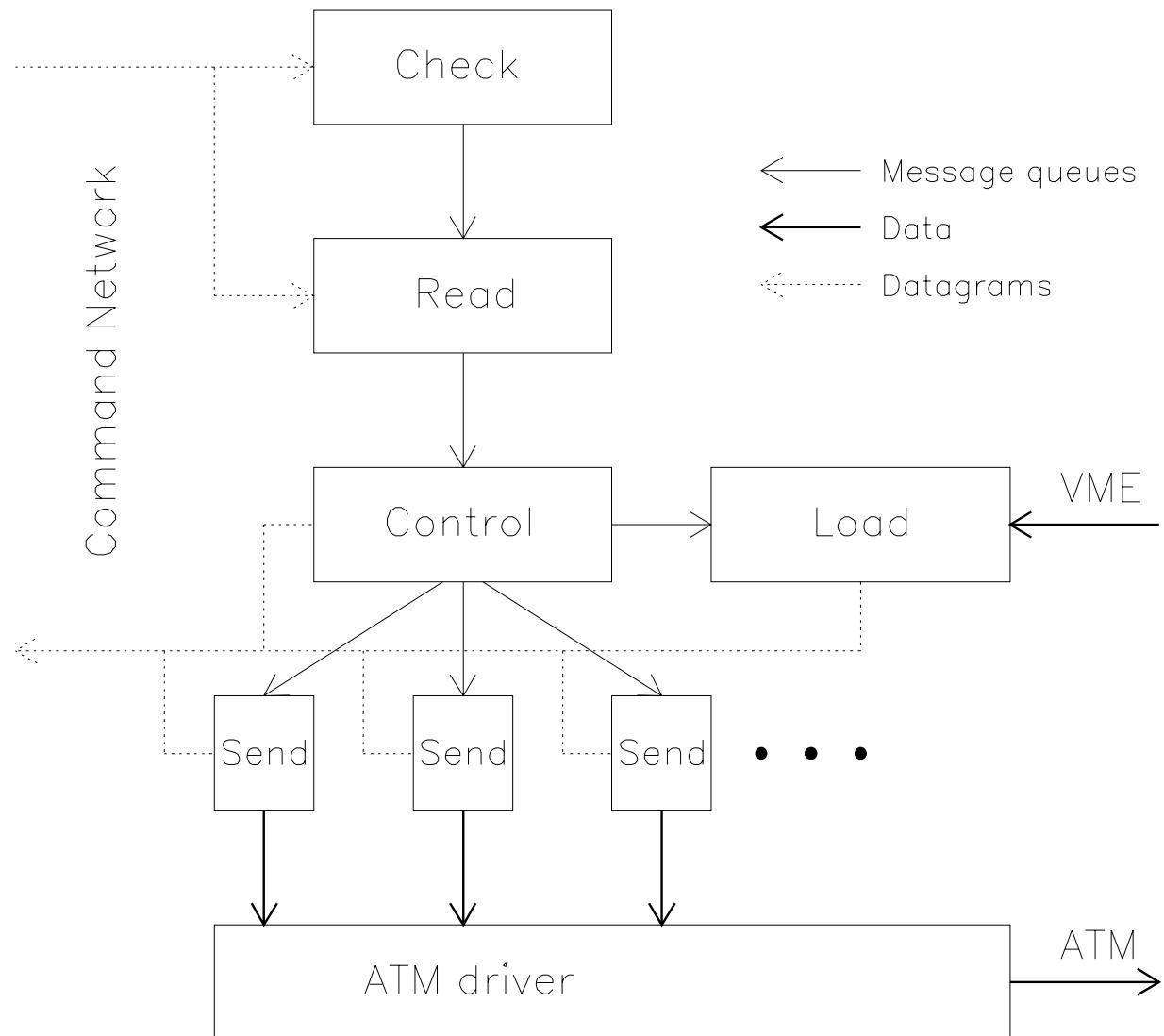
Synchronize SCPUs and L3
Listen to Trigger Manager



SCPU code

Role

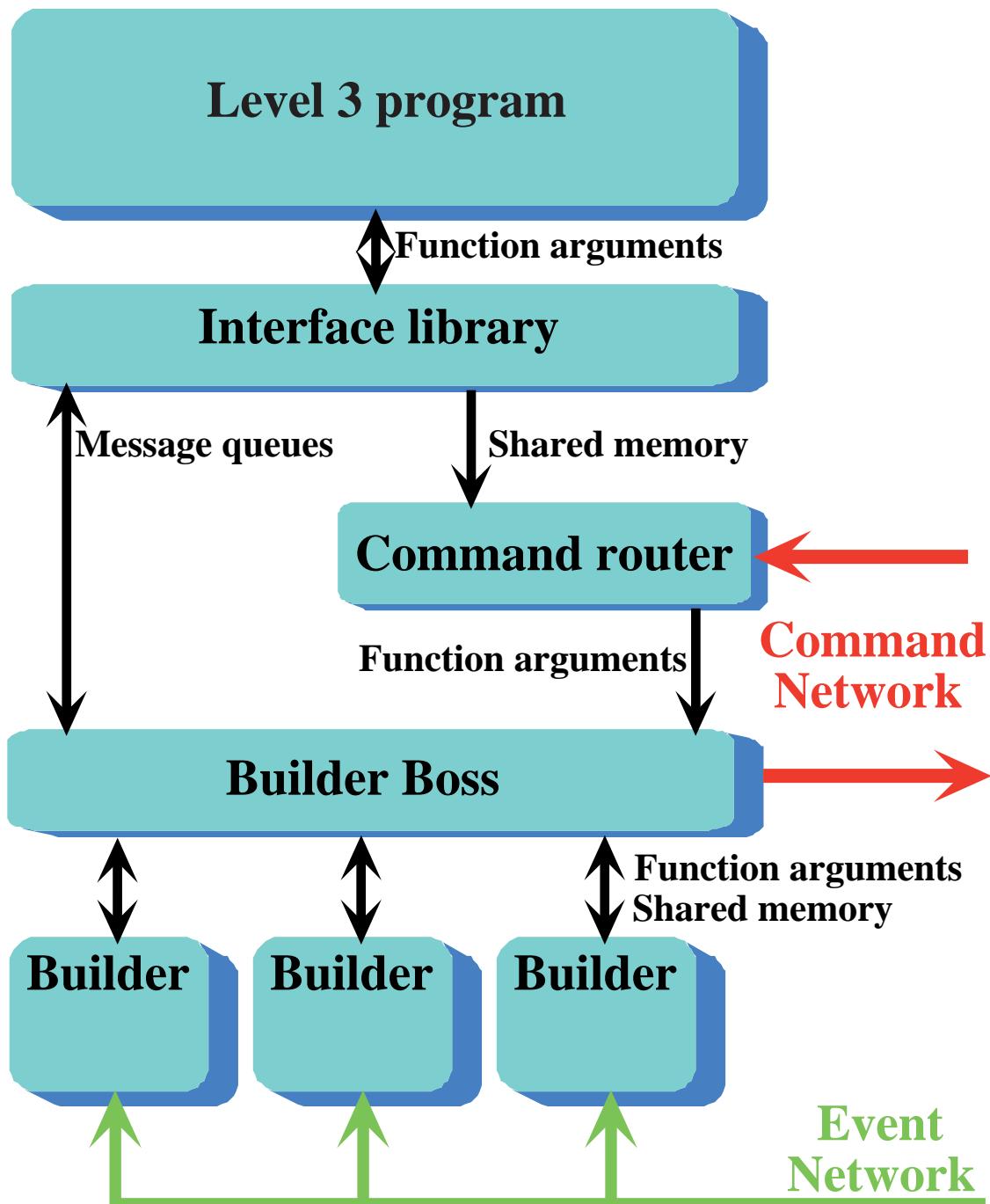
Load detector data from front end
Send them to L3



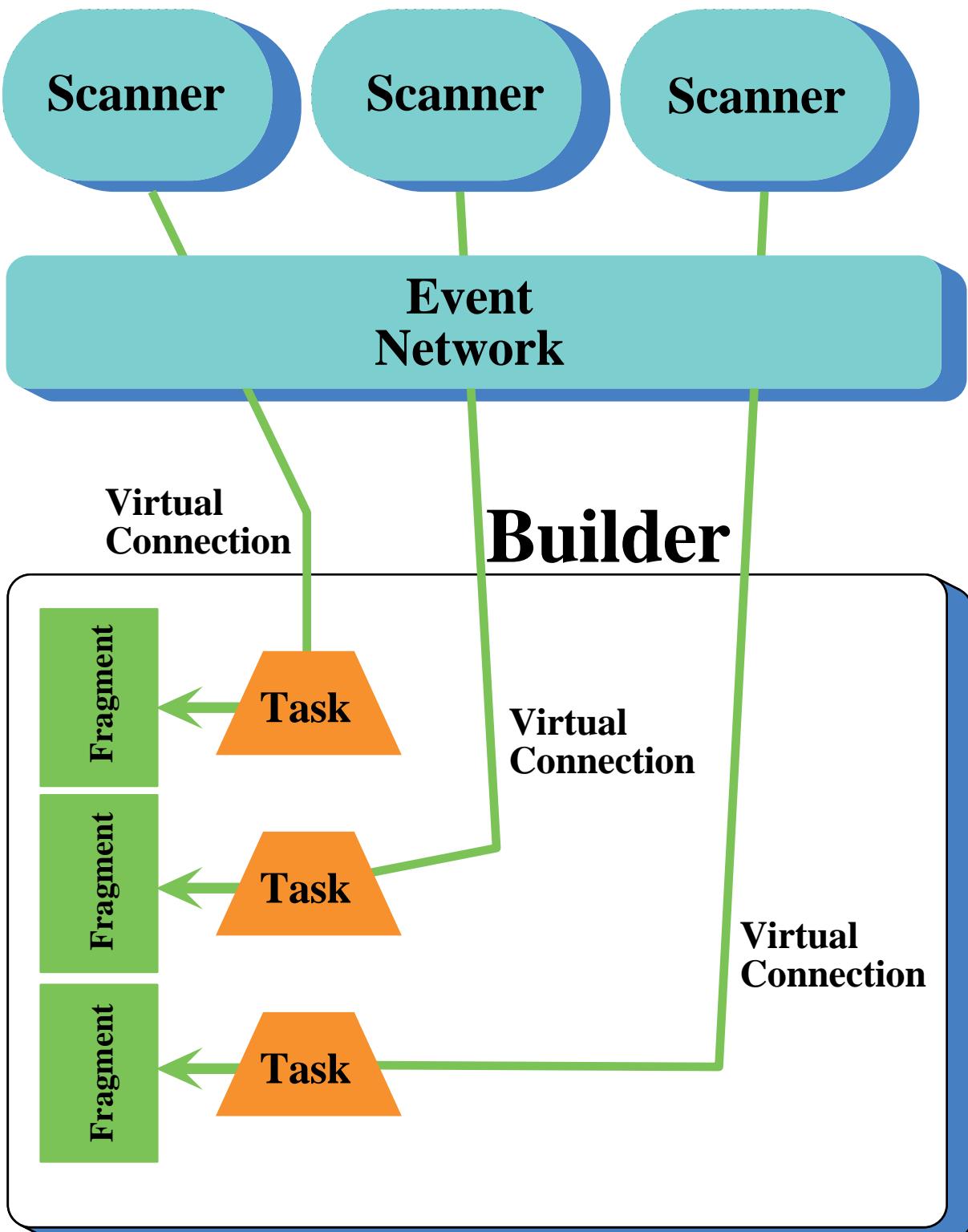
L3 receiver code

Role

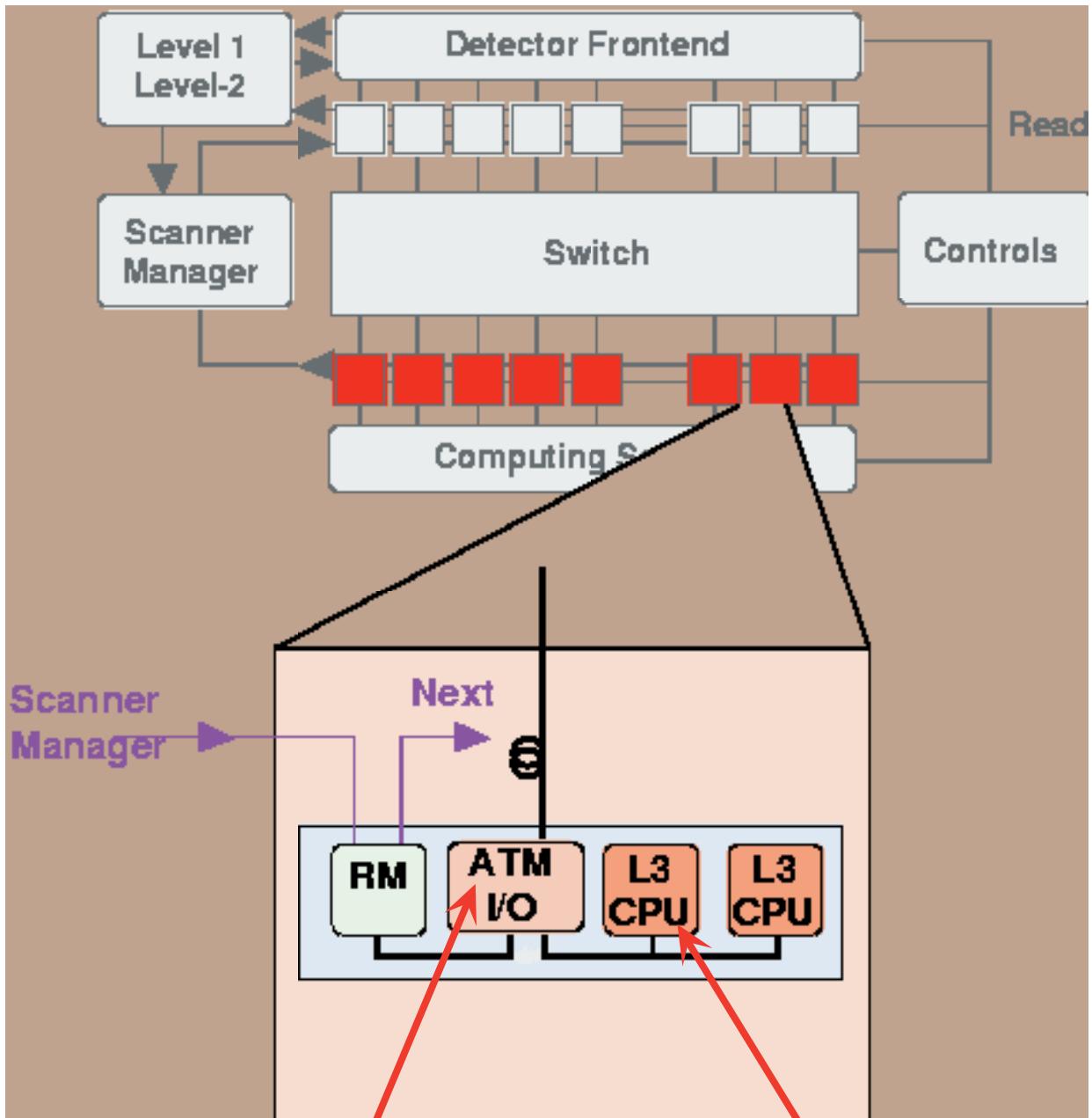
Assemble fragments into events
Give events to L3



L3 receiver code



Level 3 code



Special L3 program
Read ATM and forward
events to compute nodes

Receive from Ethernet
Full L3 processing

Conclusions

Conclusions

ATM event building works

Hardware installed

Software ported from Run I system

Some changes in algorithms

Components are integrated

Next talk is on performance of the whole

VME bandwidth needs improvement